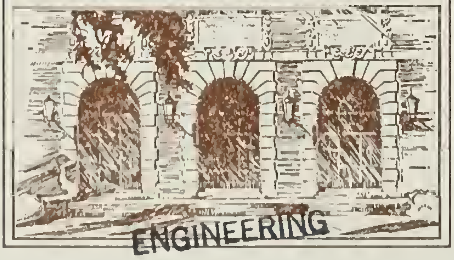


Center for Advanced Computation

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
URBANA ILLINOIS 61801

CAC Document No. 2

A STATISTICAL SYSTEM FOR ILLIAC IV

by

Stewart A. Schuster

December 11, 1970

CAC Document No. 2

A STATISTICAL SYSTEM FOR ILLIAC IV

by

Stewart A. Schuster

Center for Advanced Computation
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801

December 11, 1970

ABSTRACT

The ILLIAC IV Statistical System will be designed to take advantage of the impressive computing power of the ILLIAC IV hardware and at the same time make this power easily available to users outside the Computer Science disciplines. It is designed to exist within the framework of the ILLIAC IV Information Management and Analysis System (CAC Document No. 1) and it obeys the conventions of input language and data handling required by that system. The Statistical System is essentially a set of standard, relatively independent statistical applications programs which are interlinked through intermediate matrix files and a common control language.

ACKNOWLEDGEMENTS

First, I would like to thank Professor Daniel Slotnick for sponsoring and encouraging my work in this area.

Many of the ideas expressed within this document were originally presented to the ILLIAC IV Project in February 1970 by Dr. James L. Parker, who is currently at the University of British Columbia. I wish to express my gratitude for his continued counseling and encouragement. Also, I wish to thank Professor Michael Sher, Peter Alsberg, and Thomas Mason, who are members of the Center for Advanced Computation, for their patient reviewing and helpful suggestions given to this document. Finally, my thanks to the entire secretarial staff and especially Mrs. Shirley Brown, Mrs. Coni Allen, and Mrs. Pat Stippes for outstanding clerical assistance.

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

The Center for Advanced Computation is proposing a total computational system which will embody statistical computations, information retrieval, linear programming, modeling and simulation techniques, and a differential equation solver system. This system will be tied together by the philosophy of its input control languages and the manner in which data structures are manipulated. The proposal contained in the following pages is for the design and implementation of a Statistical System which would be imbedded in the ILLIAC IV Information Management System.

The ILLIAC IV computational capability is particularly applicable in the statistical area because most statistical computations involve a series of calculations carried out on several sets of similar data items. These computations are usually independent of one another. This means that they can be done simultaneously. This principle of simultaneity of computation is the essential design feature of the ILLIAC IV. It is expected that the parallel computational ability of the machine can yield elapsed times for computations on the order of 50 to 100 times faster than the fastest current systems for statistical operations.

The techniques themselves are relatively standard and are now fairly well known. These techniques follow the experience of the SOUPAC group at the University of Illinois on the IBM 7094 and the 360.

The system will be designed to be controlled as a sequential flow of computations with intermediate results specified at each step. A given step would be specified by indicating the statistical routine name, the source of its input or inputs, a series of parameters associated with the computation, and an indication of the various outputs and how they are to be named.

The set of statistical routines and the control language are to be designed to be broad enough to handle the majority of standard computation techniques which are found in physical, biological, and social science areas. The system will also be designed to be flexible enough so that additional techniques may either be constructed from the parts which are already available in the system or may be added either as temporary or permanent parts to the system.

The Statistical System will be controlled through the Information Management System, which will be oriented toward the retrieval of large data bases. Thus, users will not have to provide for the reintroduction of their data to the machine at each run, and file manipulation will be straightforward.

It is a primary goal that the system will be usable to researchers outside the computer science area. Thus, the standard operations will be particularly simple to avoid the requirement for learning at an unnecessary level of detail just to use the system.

2. THE OVERALL SYSTEM DESIGN

Internal to the Statistical System will be a variety of subroutines which allow the individual programs to access input matrices, make various statistical calculations, and to generate output. The formats of the input and output matrices will be sequential files, each record of which represents a row of a matrix. There will be a name and size label associated with each matrix. There will be also the option of having a vector of names associated with the columns of the matrix. Each row may also begin with a name. The presence or absence of this name will be indicated in the labeled record.

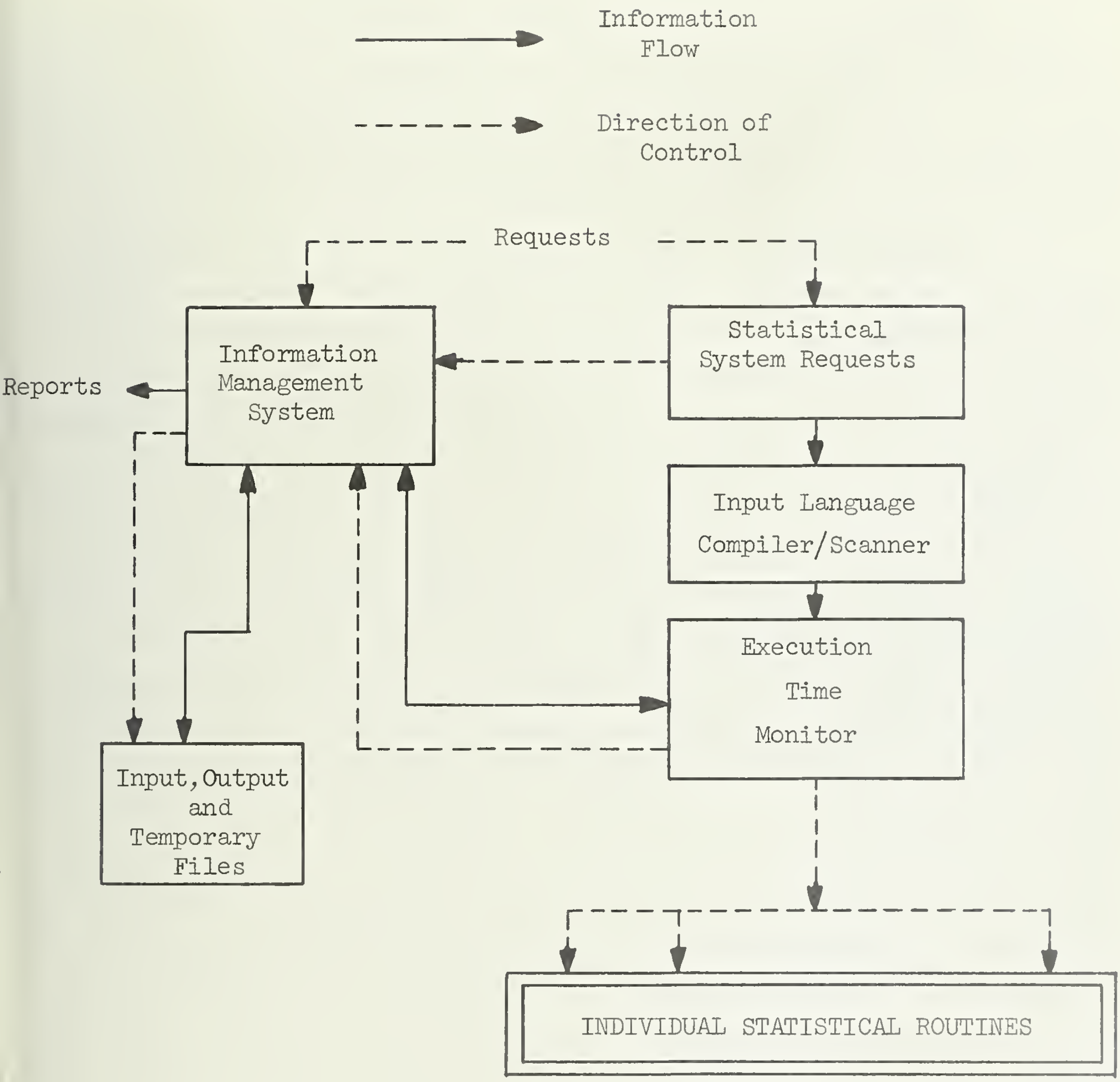
In the general language, matrix names would be considered as variables. These would be converted at execution time to file names on the disk. The system execution monitor would maintain a symbol table for these matrix names and their locations. A symbol table is a table which contains all the input and temporary alpha-numeric file names and other relevant data. Any matrices called for which were in the memory hierarchy of ILLIAC IV would be brought in by the Information Management System (IMS). A description of this system is found in reference [3]. If necessary, these files would be reformatted by the IMS for their use in the Statistical Package. It will be assumed by this system that all external files would be on the ILLIAC IV disk once execution of the problem program began. This would be done by using the Information Management System before calling the Statistical System.

In the overall language specifications, there will be no control card which indicates that one is passing into the Statistical System and out of the Information Management System. The reason is that the user need not be aware that a specific routine, say Principal Axis Factor Analysis, is contained in the statistical subsystem. Also, it should be pointed out that provision is to be made within the control language to output individual variables from each statistical program as well as matrices. These variables could be tested to indicate conditions which occurred in the computation. Within the control language these tests could cause changes in the computation sequence. For example, an F test may control a multiple regression step, or a variance limit may control the number of iterations of an iterative factor analysis.

Each routine will be implemented separately. There will be several standard matrix computation routines provided to the Statistical System designers as well as to other systems in the form of a utility package. These utility routines can be called, for example, internally from the statistical routines so that the designer of a regression routine would not have to be concerned with creating matrix multiplication codes.

Other functions available to implementors of each routine will be: ROWINPUT, ROWOUTPUT, COLUMNINPUT, COLUMNOUTPUT, and EXIT. EXIT indicates the completion of the computation and returns control to the statistical monitor. Internally, these functions work on 16 by 16 blocks. These are connected either left to right for work on rows or from top to bottom for operating on columns. This method of dynamically allocating disk and PE memory to matrices has already been devised (See Appendix A). It should be mentioned that with the data blocking scheme in 16 by 16 blocks, it is not necessary for an individual statistical program within the statistical system to be aware of the relation between the size of the matrices involved in a computation and the core size of the ILLIAC IV. If the problem fits in core, this scheme will allow it; otherwise, the partitions of the matrix will be present as needed.

Figure 1 presents a diagram of the control and information flow of the Statistical System.


Information Flow
Direction of Control
Requests
Information Management System
Statistical System Requests
Reports
Input Language Compiler/Scanner
Execution Time Monitor
Input,Output and Temporary Files
INDIVIDUAL STATISTICAL ROUTINES

Control and information flow of the statistical system

Figure 1

3. USING THE STATISTICAL SYSTEM VIA THE INFORMATION MANAGEMENT SYSTEM

There are two points which should be emphasized in this discussion of the general concepts and techniques of the ILLIAC IV Statistical System. The first of these is the ease with which researchers would be able to use this system. The second point is to demonstrate the unique capacity of the ILLIAC IV to operate with these particular statistical techniques.

Perhaps the easiest way to illustrate the use of the ILLIAC IV Statistical System would be to describe a hypothetical research problem and indicate the steps needed in order to solve it. Let us assume that a researcher has collected data on 300 variables affecting the social and economic structure of a suburb of a large city. Assume that he is interested in studying the forces which change the nature of this area. Perhaps he would begin his study by generating a correlation matrix and a factor analysis of these variables. First he would have to prepare his data in machine readable form through the Information Management System (IMS). If the data existed within several files of the IMS then the data would easily be extracted and sampled and a new file containing the matrix would be prepared to be compatible with the Statistical System conventions. At this point the matrix would be ready to be operated on by the Statistical System as a data matrix. The IMS allows the researcher to manipulate his matrix--break it into subsections either by variables or by observations by simply giving commands in the Control Language rather than by manipulating card images or by dividing up data images on tape. The researcher would then give the commands in the Control Language to do a correlation. The output of this correlation program would again be a standard matrix form which can be retained by the system. He would then specify that this output matrix be used as input to the factor analysis program. He could have the option of retaining the factor analysis matrix for further computation outputing the results of the factor analysis. At this point the researcher might investigate the printed output he has received from the factor analysis program.

It should be noted that all of these operations could have been done with a single set of instructions to the Statistical System in one pass through the machine. With the high computational capacity of the ILLIAC IV, as complex an operation as this could be done with relatively short turnaround time. This allows the research to proceed forward at a rapid rate in an "interactive" mode. The IMS allows for input errors to be easily corrected. This minimizes the long-term delays of weeks or months now experienced by researchers attempting to input observational data.

It is possible that the researcher might decide to take another set of observations at a later time in order to discover any forces of change present in the data. He might also decide to input data on a continuing basis to update his original data matrix. With the IMS he could input these observations on a continuing basis and have them merged with his original data matrix by the machine. This minimizes the data handling by the researchers. It also allows him to specify how he wants his data to be manipulated without ever having to learn the detailed method by which the computer does these operations. It is also possible that he might want to relate his observations with data from a large standard data base, such as the census data. He could simply specify the variables on which his data were to be matched with the census data, probably location, and which census data were to be incorporated in this data matrix. He could then use this expanded data matrix, which might have been expanded by several hundred variables, in the same computational process that he used before.

4. THE STATISTICAL PROGRAMS

A) Introduction

The following list specifies only those routines to be implemented as a direct result of this initial proposal. As new requirements are generated it will be quite easy to add new analytic techniques by simply obeying the conventions specified for this initial set. This feature will allow future users to broaden the base of tools originally made available.

B) The Individual Routines

The proposed statistical programs are as follows:

1. Correlations:

This program generates a product-moment correlation matrix with associated outputs. The input to this program is a standard data matrix with the rows representing observations and the columns representing variables. Effectively these columns are standardized when the matrix is premultiplied by its transpose. The computational algorithm involves adding a vector of ones to each column and multiplying each row by its transpose to form partial sums. These are then scaled at completion of the computation. Outputs are the cross-products matrix, the covariance matrix and the correlation matrix. The output matrix is built up in core as a lower triangular matrix to maximize the size of the in-core computation. The vectors of means and standard deviations are by-product outputs of this process, as is the matrix of simple linear regression coefficients.

2. Multiple Regression:

The input matrix is taken from the Correlation Program outputs. It is assumed that the dependent variables are on the right side and bottom of the input matrix. The inverse of the independent variables matrix is computed and the coefficients are computed simultaneously for each of the dependent variables. Outputs are the linear coefficients for the prediction, the multiple correlation coefficients, the inverse matrix, and an indication variable denoting the row on which singularity occurred, if it occurred during inversion. Due to the matrix blocking method described earlier, the program is not dependent on the size of the input.

3. Principal Axis Factor Analysis:

The purpose of this computation is to generate a matrix, F, which has n rows (where n is the number of variables in the study) by f columns, which is some integer smaller than n such that FF' = R where R is the intercorrelation matrix and outputs the factor matrix. The Jacobi method is used. If only eigenvalues and eigenvectors are required, the input matrix may be any real matrix. Single variables output by this program are the number of factors and the per cent of variance accounted for by those factors.

4. Varimax and Oblimax Rotations:

There are schemes for rotating factor matrices output from principal axis solutions. The first is an orthogonal method which inputs the factor matrix and outputs the rotated factors and the rotation matrix. The second technique is an oblique rotation of the factor structure which has the same input and outputs and also has a factor intercorrelation matrix.

5. Standard Scores:

This program inputs an observation matrix and outputs the means and standard deviations of each of the columns. It also outputs a scaled data matrix of the observation which has a mean of zero and a standard deviation of one for each of the columns. There is a single variable output which indicates whether any of the standard deviations are zero, indicating that one of the input variables was a constant.

6. Matrix Operations:

This program is a series of operations tied together. Each operation is referenced in the control language as though it were a separate program in the system. The following is a list of the operations and their inputs and outputs. See Appendix B for a discussion of the matrix computations routines that will be provided as a utility to use by this system and other systems.

1) Matrix addition - two inputs, one output, no variables out.
2) Matrix multiplication - two inputs, one output, error if not conformable.
3) Matrix transposition - one input, one output, no variables out.
4) Matrix inversion - one square input, one output, one variable indicating row on which matrix was singular, if at all.
5) Column delete - one input, one output.
6) Row delete - one input, one output.
7) Constant - single number input, either full or diagonal matrix output.
8) Diagonal - makes a row vector form a diagonal of the input matrix.
9) Element multiply and element divide - two inputs, one output, divides or multiplies elements of second matrix into first matrix.
10) Horizontal and vertical augment - multiple input, one output, glues matrices together.
11) Identity - generates an identity matrix of a specified size.
12) Vector - makes a diagonal matrix of a vector.
13) Move - moves one matrix to another.
14) Partition - slices a matrix into arbitrary chunks.
15) Permute - permutes rows or columns of a matrix.
16) Scalar - multiplies a matrix by a constant.
17) Subtract - subtracts second matrix from first.

The bulk of these operations are standard or have been found to be useful by the SOUPAC group at the University of Illinois.

7. Analysis of Variance and Covariance:

The design and implementation of a completely new and general Analysis of Variance system is a difficult project. Therefore, this report recommends that the BALANOVA 5 system in the SOUPAC system (see Reference 1) be used as a first model. It is applicable to a wide range of balance designs and will approximate the least squares solutions in the event that the number of applications in each cell is not proportional. It produces the least squares solutions in proportional or equal replication designs. It also has a broad coverage of the standard designs as could be delivered in a single program. This system also has the benefit that it has been fairly widely used, resulting in a fair test of its flexibility and a broad base of experience with its form of approach.

8. T-Test:

This program inputs an observation matrix and, in some cases, a vector of means and standard deviations from a previously analyzed population. It compares the input variables either in pairs or in all combinations and produces t-tests of deviations from specified means of arbitrary populations or from means of other analyzed populations.

9. Autocorrelation:

This program computes autocorrelation coefficients for an arbitrary number of variables on a series of time lags and also computes the power spectrum coefficients to give a harmonic analysis of the variables as a function of time.

10. Step-Wise Multiple Regression:

The essential process here is the same as the multiple regression program described above. The difference is that an estimate is made of the contribution of each variable in the independent set to the prediction of the dependent variable. At each step one variable is added to the independent set. The choice is the independent variable which most improves the least squares curve fit. This yields not only a set of predictors but also ranks them according to their contributions. If, at a later stage in the computation, it is seen that a variable is no longer significant it will be removed from the computation. This program necessarily operates on only one dependent variable. The outputs are the same as the multiple regression program plus a trace of the computation process. The user of the program controls the F level at which variables are to be entered or ejected from the computation process.

11. Classification:

The classification program is designed to determine group membership of an individual on a probabilistic basis. The group structure is output from a previously executed discriminant analysis program (See 12 below). For each individual the Chi Square and probability of membership in each group is given. This matrix is output.

12. Discriminant Analysis:

The purpose of this program is to give a function which will allocate a set of p variates into k different populations. The strategy is to maximize the ratio of the between-group variance to the within-group variance. The outputs are the classification vectors, group means, and dispersion matrices. The result is an eigenvalue, eigenvector solution. Specific references on the computational technique are found in the SOUPAC manual (Reference 1).

13. Frequency Analysis:

The purpose of this program is to produce generalized frequency tables and Chi Squares along requested dimensions. These frequency tables are output for other uses, the first n variables in the row being the n-1 control variables and the column being the last control variable. For each table a set of control variables is specified and for each control variable a set of values, which are to be ignored, is specified. There is also a list of variables, values, and boolean conditions which must be met for any observation to be entered into the frequency counting. This last feature is particularly useful for large scale complex data bases such as those handled by the Information Management System. Input is a standard observation matrix. Each row of each table is allocated separately so that rows are allocated in storage only for those elements which actually occur in the data. This means that the limits of the data do not have to be specified by the user and that the maximum number of tables is processed on each pass of the data. The input parameters are in a block structure in the sense that within one set of boolean conditions a wide variety of tables may be indicated. It is possible that on the first pass, while only one block is being processed, the program can range all the variables so that on subsequent passes it can determine the maximum amount of core required for a given set of tables, and thereby do more than one block per pass on multiple passes--otherwise the program would have to make one separate pass simply to range the data. For large data bases an option may be inserted that allows the user to specify the ranges of the data items.

14. Transformations:

The transformations program is similar to the matrix program except that its basic element of operation is a data row rather than a whole matrix. A set of codes for the transformation program represents a series of row operations which is repeated for each row of the input matrix. There is also a set of operations which are executed just once after the total matrix has been passed by the program. At the completion of the basic set of instructions the current row, as modified by the user, is read out and the next row is read in. It is also possible to cause the output of special rows into other matrices during the processing of the basic matrix. There is also a set of row elements which are carried forward and not zeroed out between the processing rows. The basic transformations are done between elements of a given row to produce new elements or to replace old elements of that row. These instructions also have the capability of branching to symbolic labels based on tests of certain variables within a given row. The types of operations that are common are ADD, SUBTRACT, RECODE, PERMUTE, etc. A complete list of the set used by the SOUPAC system is to be found in the SOUPAC manual (Reference 1). This program is written as a table-driven structure with each operation as an entry point from a larger computed GOTO (or CASE statement as in ALGOL) so that the number of operations is essentially unlimited once the basic structure is present.

REFERENCES

[1] "SOUPAC Statistically Oriented Users Programming and Consulting", DCS Report No. 370. Urbana, Illinois: Department of Computer Science, University of Illinois at Urbana-Champaign, (December 1969).

[2] Chouinard, P. "Outline of a blocking scheme for implementation in a GLYPNIR written statistical system", Memorandum. Urbana, Illinois: ILLIAC IV Project, University of Illinois at Urbana-Champaign, (January 9, 1970).

[3] Schuster, Stewart A. "An Information Management and Analysis System for ILLIAC IV", CAC Document No. 1. Urbana, Illinois: Center for Advanced Computation, University of Illinois at Urbana-Champaign, (December 11, 1970).

[4] Sameh, A. "On Jacobi and Jacobi-like Algorithms for a Parallel Computer", Journal of Mathematics of Computation, (July 1971) (in press)

[5] Parker, James L. "The ILLIAC IV Statistical System", Proposal submitted to the Graduate College by the ILLIAC IV Project, University of Illinois at Urbana-Champaign, (February 1970).

APPENDIX A

OUTLINE OF A BLOCKING SCHEME FOR IMPLEMENTING A STATISTICAL SYSTEM [2]

For a statistical system of any significance to be written for ILLIAC IV, it is essential that the various programs be able to handle non-memory contained arrays. This is not the same insurmountable problem which occurs in the compiler area, since one knows ahead of time how an array is going to be referenced. Also, one has direct control of data overlay by explicit I/O statements because the programmer is both the user of the system and the designer of the data flow through the system. In this way, he can monitor most of the activities.

For most statistical applications, array referencing is by rows and, less frequently, by columns. The purpose of this appendix is to propose methods of blocking an array such that rows and columns are easily accessible for use within a statistical system. This particular presentation is for an arbitrary statistical program written in GLYPNIR which has one n by n matrix to be handled. The extension to several matrices is straight forward.

A solution is to block the array as in Figure 2 into 16 by 16 blocks. Figure 2 considers an 80 by 80 data matrix. For example, the problem program may determine to do a row operation on the i^{th} row of a data matrix (hereafter referred to as A). The problem program would always do arithmetic in a memory block which is at least large enough to contain 16 rows or columns of A. This work buffer shall be referred to as BUFFER1. If the i^{th} row is in BUFFER1, processing proceeds. If the i^{th} row is not in BUFFER1, the problem program determines whether or not it needs the current contents of BUFFER1. The problem of saving work buffers is discussed later. The problem program can then call a GETROW routine which indicates the following:

1. It needs rows (by default of calling GETROW).
2. Return the i^{th} one.
3. It is within matrix A (in case there is more than one matrix in the program).
4. Address of BUFFER1 (there may be BUFFER2, BUFFER3, etc.).

The GETROW subroutine looks to see what blocks it has stored in memory. If the required blocks are in memory, the problem program gets the blocks directly (GETROW moves the data from its save area to BUFFER1, see Figure 3). Any additional blocks needed can be secured by sending a request to the Information Management System (IMS). IMS can be given a list of blocks needed and where to put them in ILLIAC memory. Similarly, PUTROW saves the blocks.

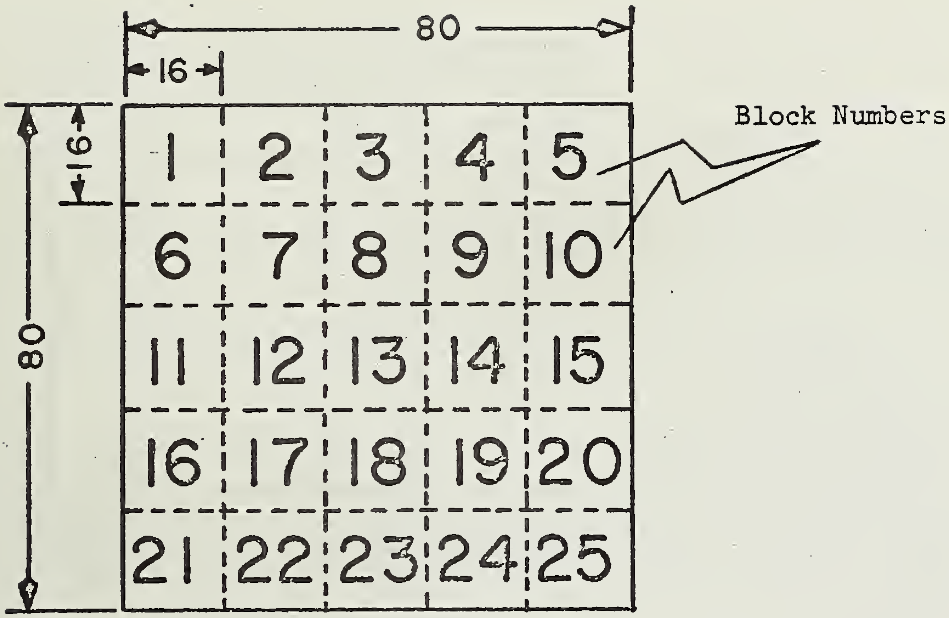
The use of GETCOL and PUTCOL is also similar to GETROW. The trick is that within BUFFER1 the data is worked on as rows, see Figure 4. It is the job of GETCOL AND PUTCOL to get the column into BUFFER1 in row form. There are two basic solutions to this problem.

First, within each 16 by 16 segment, data is stored "straight". See Figure 5 for a diagram of straight storage. For rows, no remapping is done. Data is copied directly by GETROW into BUFFER1. For columns, each 16 by 16 block is transposed and the blocks are lined up side by side. It should be possible to transpose four such 16 by 16 blocks at a time.

The second solution would be that within each 16 by 16 segment, data is stored "skewed". See Figure 5. For rows, a row is brought up to be routed back into "straight" alignment and then stored. Columns may be "indexed out", routed, and stored into BUFFER1 as rows. Note that routing in both cases implies a 16 PE end around route which is not particularly difficult to implement.

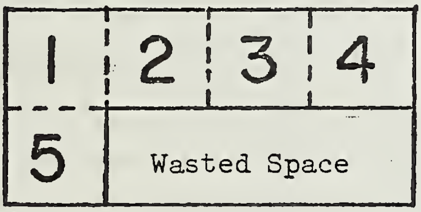
The trade-off between straight and skewed within the 16 by 16 blocks is that with straight one has no remapping for rows but a fairly cumbersome transposition remapping for columns. With skewed one has essentially similar relatively straightforward remappings for both row and column access. The outstanding question is, "What is the ratio of row accesses to column accesses?"

Two more routines would probably be useful in the repertoire of data retrieval routines, namely GETDIAGONAL and PUTDIAGONAL, for handling the main diagonal of an array. It is clear that the skewed method of storage complicates the retrieval of the main diagonal. It should also be noted that the disk is used only if the GET and PUT routines can't store the data in their own internal save areas.



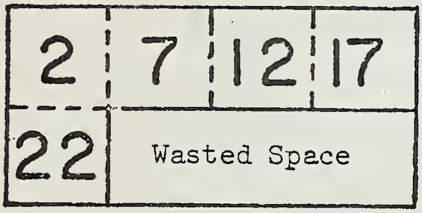
Blocking structure superimposed on an 80 by 80 data matrix

Figure 2



Mapping of blocks into ILLIAC memory for accessing rows 0-15

Figure 3



Mapping of blocks into ILLIAC memory for accessing columns 16-31

Figure 4

	PE_i	PE_{i+1}	PE_{i+2}	. . .	PE_{i+15}
longword i	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$		$a_{1,15}$
i+1	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$		$a_{2,15}$
	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$		$a_{3,15}$
i+15	$a_{16,1}$	$a_{16,2}$	$a_{16,3}$		$a_{16,16}$

Straight Storage

	PE_i	PE_{i+1}	PE_{i+2}	. . .	PE_{i+15}
longword i	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$		$a_{1,15}$
i+1	$a_{2,15}$	$a_{2,1}$	$a_{2,2}$		$a_{2,14}$
	$a_{3,14}$	$a_{3,15}$	$a_{3,1}$		$a_{3,13}$
i+15	$a_{16,2}$	$a_{16,3}$	$a_{16,4}$		$a_{16,1}$

Skewed Storage

Storage schemes for 16 x 16 matrices

Figure 5

In summary:

1. Divide the matrix into 16 by 16 blocks. These are canonical units which map well into disk segments.
2. Three GET and three PUT routines which keep track of the data blocks upon command by the problem program. These six routines are all entries in the same subroutine. If that isn't possible, it can be made one subroutine with six options.
3. Data is stored in memory in 16 by 16 blocks if possible. After available memory is used, write the rest out on disk.
4. Whether conceptually the data is rows, columns, or diagonals the ILLIAC IV is a row machine, and should be used that way. The GETPUT routines will perform the mappings into row order.
5. All arithmetic is done in work buffers.
6. Write it in GLYPNIR or COCKROACH.

A few additional points should be made clear. The GET and PUT routines try to save blocks in memory first. If the blocks don't fit, they are written out onto disk. This function is transparent to the statistical programs, themselves. It is possible, therefore, to write the statistical programs using temporary GET and PUT routines which only store in memory. After the statistical programs are written, expansion of GET and PUT to handle non-memory contained matrices, greatly enhances the power of the statistical system, and may be done without changing the original statistical code.

The discussion in the first part of this paper assumes that a statistical program needs buffer space at least large enough to handle 16 complete rows or columns of the data matrix. This potentially represents some upper bound on the size of matrices the system can handle. The system could perhaps be designed to operate on only four 16 by 16 blocks at a time in a standard 16 by 64 work buffer. This would guarantee the facility of working on problems where it is not always possible to contain 16 complete rows or columns in memory. Implementation of such a scheme increases control-type-statement overhead and flexibility.

APPENDIX B
MATRIX COMPUTATIONS ON ILLIAC IV [4]

Since matrix computations, such as multiplication and inversions, are necessary utilities for several different applications, they should be provided to all users and system designers in the form of a subroutine package. Part of such a package is currently being implemented and a proposal to complete this work has been submitted to the Advanced Research Projects Agency.

However, several other matrix operations are required in the context of a Statistical System and thus would be provided as part of this system. These routines are listed and explained in section 4.B.6. Also listed, depending on the generality of input forms of the matrices, are some routines that may have the same names as those provided as utilities. This is necessary since a data matrix may be partitioned in any form for the Statistical System. Each partition's data may exist as a separate record or parts of several records in files within the Information Management System. It may be very inefficient to produce the concatenation each time to create one record for the matrix input required for the utility matrix computation routines. Since the partitions may be needed in later computations, it would be disadvantageous to destroy their forms. It should be understood that the duplicated statistical routines only contain algorithms to find the partitions in the sequence required by the calculation. Once the proper partitions are found the routine will call the utility routine to execute the calculation. New partitions are then found and the utility routine is called again until the computation is complete.

The discussion implies that although a matrix multiplication routine may be provided to all users, we will also need a driver routine which finds the data as needed in the calculation. It then calls the provided routine to perform the actual sub-calculations.

The routines that would be provided as a utility to the Statistical System designers are listed:

1) Matrix Multiplication, Addition, and Subtraction
2) Square Root

3) Matrix Inversion

4) Eigen Vectors

5) Eigen Values

6) Evaluation of Determinants

7) Matrix Transposition

[illegible]

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

RIGINATING ACTIVITY *(Corporate author)* enter for Advanced Computation niversity of Illinois at Urbana-Champaign rbana, Illinois 61801	2a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED 2b. GROUP

EPORT TITLE

STATISTICAL SYSTEM FOR ILLIAC IV

ESCRIPTIVE NOTES *(Type of report and inclusive dates)*
esearch Report

UTHOR(S) *(First name, middle initial, last name)*

tewart A. Schuster

EPORT DATE ecember 11, 1970	7a. TOTAL NO. OF PAGES 30	7b. NO. OF REFS 5
CONTRACT OR GRANT NO. SAF 30-(602)-4144 PROJECT NO. RPA Order 788	9a. ORIGINATOR'S REPORT NUMBER(S) CAC Document No. 2 9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*	

DISTRIBUTION STATEMENT

opies may be requested from the address given in (1) above.

SUPPLEMENTARY NOTES one	12. SPONSORING MILITARY ACTIVITY Rome Air Development Center Griffiss Air Force Base Rome, New York 13440

ABSTRACT

The ILLIAC IV Statistical System will be designed to take advantage of the impressive computing power of the ILLIAC IV hardware and at the same time make this power easily available to users outside the Computer Science disciplines. It is designed to exist within the framework of the ILLIAC IV Information Management and Analysis System (CAC Document No. 1) and it obeys the conventions of input language and data handling required by that system. The Statistical System is essentially a set of standard, relatively independent statistical applications programs which are interlinked through intermediate matrix files and a common control language.

14. KEY WORDS	LINK A		LINK B		LINK
	ROLE	WT	ROLE	WT	ROLE
Mathematics of Computation (General)					
Social and Behavioral Sciences (General)					